Системная разработка игр на Unity 3D



Георгий Агасарян

Георгий Агасарян

Системная разработка игр на Unity 3D

«Автор»
2026

Агасарян Г.

Системная разработка игр на Unity 3D / Г. Агасарян — «Автор», 2026

Знать Unity — ещё не значит уметь выстроить разработку целой игры. Когда скрипты, механики и сцены должны стать единым проектом, на первый план выходят проектирование, архитектура и последовательность решений.«Системная разработка игр на Unity» — книга для тех, кто уже освоил движок и готов создавать целостные игровые проекты. Она проводит читателя от геймдизайн-документа и декомпозиции задач до реализации механик, создания локаций и объединения всех элементов в согласованную систему.В центре внимания — связь идей и кода: как превратить идею в план работы, определить связи и взаимодействие игровых систем, организовать структуру проекта и учесть ограничения. Вы узнаете не только как реализовать задуманное, но и как выбирать решения, проверять их на практике и закладывать возможности для развития игры.Это не знакомство с интерфейсом Unity и не сборник разрозненных рецептов, а системный подход к созданию игры, в которой механики, код и локации работают на общий результат.

© Агасарян Г., 2026
© Автор, 2026

Георгий Агасарян
Системная разработка игр на Unity 3D

Глава 1. Создание дизайн документа

Дизайн документ это...

Дизайн-документ — не сочинение о будущей игре, а рабочая система решений. Дисдок помогает команде или одному разработчику, одинаково понимать цель, правила, ограничения и критерии готовности. Книга построена от общего к частному: сначала вы формулируете видение, затем описываете игровой цикл и механики, после этого связываете дизайн с реализацией в Unity 3D и производственным планом.

Результат обучения

Сформулировать идею: аудитория, платформа, фантазия игрока, уникальность и ограничения.

Построить игровой цикл: понять, какие действия игрок повторяет и почему они остаются интересными.

Описывать проверяемые механики: через состояния, правила, параметры, исключения и критерии приёмки.

Наложить дизайн на Unity 3D: выделить данные, компоненты, ScriptableObjects, сцены, префабы и события.

Подготовить жанровый мини-GDD: для вашей игры, на примере Souls-like, стелса, песочницы, RPG, FPS или хоррора.

Рекомендуемый маршрут

Прочитайте страницы до раздела 2 и заполните одностраничный концепт.

Выберите один жанровый пример из изложенных в конце главы и замените учебные значения своими.

Соберите прототип одной петли длительностью 3–10 минут.

Проведите минимум пять наблюдаемых тестов и обновите документ, учитывая факты.

Перед производством проверьте чек-листы на последней странице главы.

Главный принцип - Документ должен уменьшать количество неоднозначных устных договорённостей. Если раздел не влияет на решение, оценку или тест — его можно сократить.

Условные обозначения

Решение — То, что команда обязуется реализовать в текущей версии.

Гипотеза — Предположение, которое нужно проверить прототипом или тестом.

Параметр — Число или настройка, вынесенная из кода и доступная для баланса.

Критерий — Наблюдаемое условие, по которому функция считается готовой.

1. Что такое GDD и из каких документов он состоит

GDD (Game Design Document) — центральное описание опыта игрока и правил игры. В маленькой команде это может быть один файл с таблицами. В большой — набор связанных страниц.

Минимальный комплект

One-pager — Что это за игра и для кого? 1 страница.
Vision / pillars — Какой опыт нельзя потерять? 1–2 страницы.
Core loop — Что игрок повторяет? 1 схема + пояснения
Спецификации механик — Как точно работает каждая система? 1–4 страницы на систему.
Контент-дизайн — Какие уровни, враги, предметы и задания нужны?Таблицы и шаблоны.
UX-спецификация — Как игрок управляет и получает обратную связь? Экраны, состояния, тексты.
Tech notes — Как это собирается в Unity?Компоненты, данные, события.
План тестирования — Какие гипотезы и метрики проверяем? Чек-лист + журнал.

Три уровня подробности

Уровень 1 — зачем: цель, эмоция, аудитория, дизайнерский принцип.
Уровень 2 — что: действия, состояния, правила, контент, интерфейс.
Уровень 3 — как проверить: числа, сценарии, телеметрия, критерии приёмки.
Ошибка новичка. Написать сюжет, список оружия и лор раньше, чем определены действия игрока и условия выбора. Начинайте с глаголов: исследовать, скрываться, целиться, обменивать, рисковать.

Правило актуальности

У каждого раздела дисдока, должны быть ответственное лицо, текущий статус и дата изменения. Устаревшее решение либо обновляют, либо явно помечают как архивное. Команда не должна угадывать, какой из двух противоречащих абзацев верен.

2. Одностраничный концепт и дизайн-столпы

Шаблон one-pager

Рабочее название — Как коротко ссылаться на проект? Например «Пепельный Предел» или «Spectre»

Жанр и камера — Как игрок видит и читает мир? Action RPG, вид от третьего лица.

Платформа — Где и каким вводом играют? ПК, клавиатура/мышь и геймпад.

Аудитория — Какой опыт у игрока? Знаком с экшен-RPG, любит имме мир.

Фантазия — Кем игрок себя ощущает? Охотник, побеждающий знанием.

Крючок — Чем игра отличается? Следы прошлых попыток меняют арену.

Сессия — Сколько длится типичный заход? 20–40 минут

Объём MVP — Что доказывает жизнеспособность? 1 хаб, 1 уровень, 3 врага, босс

Не делаем — Что сознательно исключено? Открытый мир, PvP, транспорт

Дизайн-столпы

Столп — короткое правило выбора. Он помогает решить спор между двумя хорошими идеями. Обычно достаточно трёх или четырёх.

Например:

Читаемый риск — опасность заметна заранее; наказание следует за решением, а не за скрытой информацией.

Знание сильнее числа — понимание паттернов важнее бесконечного роста характеристик.

Короткий путь после открытия — исследование мира регулярно создаёт полезные сокращения.

Проверка столпа

Хороший столп влияет на конкретный выбор. Вопрос «добавлять ли случайные мгновенные ловушки?» получает ответ: нет, если они нарушают читаемый риск.

Фраза «красивая атмосфера» слишком общая: она не задаёт критерия решения в отличие от сформулированного столпа.

Практика на 20 минут. Напишите по одному предложению, состоящему из крючка и трёх столпов. Затем удалите все прилагательные, которые невозможно проверить наблюдением игрока.

Это значит что нужно убрать из описаний предметов, локаций или персонажей все субъективные, оценочные и абстрактные слова, которые игрок не может подтвердить своими глазами или ушами прямо в игре.

Что считается «непроверяемым»?

Слова, которые описывают чувства, историю или оценку автора, но не дают конкретной визуальной или физической информации.

Субъективные оценки: «красивый», «ужасающий», «великолепный», «уютный». (То, что автору кажется уютным, игроку может показаться тесным).

Абстрактные понятия: «древний», «магический», «проклятый», «забытый». (Если на мече просто написано «древний», но он выглядит как новый, игрок не может это проверить наблюдением).

Скрытые свойства: «тяжелый» (если игрок не видит, как персонажу тяжело), «острый» (игрок не видит как им что то разрезается), «громкий» (если громкость такая же как у окружения).

Что считается «проверяемым»?

Это факты, которые очевидны из графики, звука или механики.

Цвет и форма: красный, квадратный, зазубренный.

Состояние материала: ржавый, влажный, треснувший, горящий.

Материал: железный, деревянный, шерстяной.

3. Игровой цикл и решения

Игровой цикл показывает повторяющуюся последовательность: действие обратная связь изменение состояния новое решение. Он должен объяснять, почему игрок хочет сделать следующий шаг.

Три цикла масштаба

Секунды — заметить, прицелиться, выстрелить, оценить попадание. сокращая : ввод, тайминг, feedback.

Минуты — разведать комнату, выбрать маршрут, вступить в бой, собрать ресурс, другими словами: риск, награда, вариативность.

Часы — пройти регион, улучшить сборку, открыть способность, вернуться к безопасной точке: прогрессия и мотивация

Как записать core loop

Назовите 4–7 основных глаголов игрока. Не включайте редкие меню и косметические действия.

Соедините действия причинно: ресурс, информация или изменение мира должны вести к следующему решению.

Отметьте места выбора: безопасно/быстро, тихо/громко, потратить/сохранить.

Опишите провал: что теряется, чему игрок учится, как быстро возвращается к действию.

Проверьте цикл в прототипе без сюжета и дорогого контента.

Пример: стелс-цикл

Наблюдать за патрулём > отметить укрытия > выбрать маршрут > отвлечь охранника > пройти в запретную зону > получить разведданные > открыть более безопасный выход > покинуть зону задания.

Интерес возникает из неполной информации, ограниченных инструментов и возможности исправить ошибку.

Диагностика: Если действие не меняет состояние игры и не даёт информации, оно, вероятно, декоративно или лишнее.

4. Спецификация механики: универсальный формат

Одна механика должна быть описана так, чтобы программист мог оценить реализацию, художник — определить нужную обратную связь, а тестировщик — составить план проверки.

Карточка механики

Цель — Какую проблему игрока решает механика и какой опыт создаёт.

Условия входа — Состояние персонажа, ресурс, поверхность, дистанция, доступность цели.

Ввод — Кнопка, удержание, очередь команд, приоритеты, переназначение.

Правила — Формула, этапы, окна времени, столкновения, отмена.

Состояния — Idle, Start, Active, Recovery, Interrupted и переходы между ними.

Параметры — Числа, которые балансируются без изменения кода.

Feedback — Анимация, звук, камера, VFX, UI, вибрация.

Исключения — Край платформы, пауза, смерть цели, смена сцены, низкий FPS.

Критерии приёмки — Наблюдаемые тесты, «дано это — когда это — тогда будет вот это».

Пример: уклонение

Цель: дать короткий инструмент избегания урона с ценой в выносливости.

Ввод: нажатие Dodge + направление.

Условия: персонаж на земле, жив, не оглушён, выносливость не менее 20 единиц.

Состояния: запуск 0,10 с; активная фаза 0,28 с; восстановление 0,32 с. Неуязвимость действует с 0,08 до 0,22 с.

Расход выносливости списывается при переходе в Start.

Направление фиксируется в начале, но камера может вращаться. При столкновении со стеной дистанция перемещения сокращается, а состояние не прерывается.

Критерии приёмки

При 19 единицах выносливости команда не запускает уклонение и выдаёт короткий сигнал отказа.

Удар, попавший в окно неуязвимости, не наносит урон; реакция визуальный контакт допускается только при читаемом эффекте.

После окончания Recovery снова доступны движение и атака; очередь ввода хранит одну команду не дольше 0,15 с.

Важно. Числа в примере — стартовая гипотеза, а не универсальный баланс. Объявляйте их в настраиваемых переменных и меняйте в инспекторе при тестировании.

5. Баланс, прогрессия и экономика

Баланс — управление решениями, а не поиск единственного «правильного» числа. Документ должен показывать назначение параметров, допустимый диапазон и зависимость от других систем.

Минимальный набор параметров для старта

Здоровье игрока на старте — 100 единиц это запас на 3–5 ошибок

Урон лёгкой атаки — 24 либо в диапазоне 18–32, задаст базовый темп боя

Стоимость уклонения — 20 либо в диапазоне 15–30, для того что бы ограничить спам уворотами.

Восстановление выносливости — 18/с. для возврата к активности после небольшой паузы.

Три кривые прогрессии

Мощь: урон, защита, новые комбинации. Рост не должен отменять необходимость принимать решения.

Компетентность игрока: знание паттернов, карты и синергий. Для Souls-like и стелса это ключевая кривая.

Доступ: новые зоны, типы заданий, инструменты, специальные возможности.

Экономическая петля

Источник создаёт ресурс > смерть удаляет его > хранилище ограничивает накопление > обмен задаёт ценность.

Для каждой валюты укажите: откуда она приходит, на что тратится, когда становится дефицитной и что предотвращает бесконечную инфляцию.

Пример расчётной цели

Обычный враг должен погибать от 4–6 базовых попаданий оружием соответствующего уровня.

Базовый урон 24 при здоровье врага 120 даёт 5 попаданий. Критический удар ×1,5 сокращает число попаданий, но требует определённых условий. Это делает бонус инструментом мастерства, а не обязательным условием.

Красный флаг. Не балансируйте только средними значениями. Проверяйте крайние сборки, накопление ресурсов, бесконечные циклы лечения и комбинации, полностью отменяющие риск.

6. Уровни, миссии, задания и контент

Контент-документ превращает правила игры в повторяемые единицы производства. Хороший шаблон уровня или задания снижает стоимость согласования и делает качество более предсказуемым.

Карточка уровня

Цель опыта — Научить обходить бронированного врага через позиционирование и использование ландшафта.

Входное знание — Игрок умеет уклоняться и наносить удар в спину.

Новая идея — Шумные приманки разворачивают тяжёлого врага в сторону источника звука.

Ритм — разведка, короткий бой, передышка, усложнение, развязка.

Ориентиры — башня, красный мост, кузница; видны минимум из двух точек.

Награды — короткий путь, материал улучшения, необязательный лор, валюта.**Провалы —** Игрок изучает механики бронированных врагов, учится читать паттерны новых противников.

Бюджет — 12 минут, 3 типа врагов 1 из них новый, 2 интерактивных ситуации (сбросить на врагов камень, спустить лестницу, открывающую короткий путь), 1 неочевидный уникальный момент (перепрыгнуть через провал в полу, открыть сундук).

Карточка задания RPG

Предпосылка и мотивация персонажей без скрытых знаний автора: NPC просит героя **что-то** сделать, игрок **находит записку** либо задание выдаётся автоматически после **какого-то** события».

Цель игрока в терминах действия: найти, убедить, доставить, победить, выбрать.

Состояния: не начато, активно, обновлено, провалено, завершено.

Ветвления и условия: факт мира, репутация, предмет, время, предыдущий выбор.

Награды и последствия: опыт, валюта, доступ, отношения, изменение мира, предмет.

Защита от тупиков: ситуации, когда потеря предмета или смерть NPC делают задание невыполнимым, и методы их обхода.

Например: загрузка сохранения, появление квестового предмета в продаже у торговца в случае его утраты или запрет на удаление предмета из инвентаря до достижения целей задания.

Контентная матрица

Сделай таблицу для планирования контента игры:

В строках пиши комнаты или уровни.

В столбцах указывай всё, что там будет: врагов, окружение, призы и новые правила игры.

Зачем это нужно?Так ты сразу увидишь, где в игре стало скучно (повторы), а где игрок может запутаться (слишком много всего сразу).

Как вводить новые правила (механики):

Покажи: дай игроку попробовать новинку там, где ему ничего не угрожает.

Проверь: создай простую ситуацию, где это правило нужно применить.

Смешай: добавь это правило к тем, что игрок уже знает.

Усложни: только теперь делай уровень по-настоящему трудным.

Правило производства. Любая уникальная единица контента должна оправдывать стоимость: обучать, проверять, менять темп, раскрывать выбор или создавать сильный запоминающийся момент.

7. Управление, камера, интерфейс и звук

UX-раздел описывает не красоту экранов, а передачу состояния игры. Игрок должен понимать: что произошло, почему это произошло, какие варианты доступны и что случится после действия.

Карта управления

Движение - WASD, левый стик: блокируется при жёстком оглушении.

Камера - мышь, правый стик: чувствительность раздельная по осям

Взаимодействие - E, X/Square: приоритет ближайшего видимого интерактивного объекта.

Лёгкая атака - ЛКМ, RB/R1: не срабатывает при работе с интерфейсом (инвентарь, меню)

Уклонение - Space, B/Circle: удержание не повторяет действие

Камера

Зафиксируйте FOV, дистанцию, ускорение, сглаживание, коллизии, вертикальные ограничения и поведение при захвате цели. Для каждого режима запишите переход: исследование, бой, прицеливание, диалог. Камера не должна сообщать ложную информацию, например прятать угрозу во время обязательной реакции.

Иерархия обратной связи

Первичный канал: анимация и изменение мира или состояния — дверь открылась, враг пошатнулся и обратил на это внимание.

Подтверждение: звук, визуальный эффект, импульс камеры, вибрация.

Точное значение: число, шкала, значок статуса которые игрок может точно идентифицировать.

Объяснение: это может быть подсказка, журнал, экран обучения — только когда нужно.

Пример

Обратная связь при парировании атаки в игре Bloodborne осуществляется через несколько каналов:

Звуковой: проигрывается громкий и уникальный звуковой эффект, однозначно сигнализирующий об успешном парировании.

Визуальный: у противника срабатывает анимация оглушения, которая явно отличается от анимаций атаки или получения обычного урона.

Геймплейный: цель перестает атаковать на некоторое время, давая паузу для проведения критического удара (Visceral Attack) или для того, чтобы игрок успел восстановить здоровье (подлечиться).

Первичный канал здесь — это само срабатывание механики: атака прерывается, и персонаж игрока не получает урона.

Подтверждением успеха служит четкий, уникальный звук.

Точное значение — Враг оглушен, не атакует, занял позицию позволяющую провести критический удар.

Объяснение — В Bloodborne нет прямых обучающих окон или обязательных подсказок от разработчиков, напрямую объясняющих механику парирования выстрелом из пистолета. Управление описывается лишь базово через текстовые подсказки (например, записка в Сне Охотника указывает на кнопку выстрела **L2**, но не объясняет тайминги парирования), из-за чего игроку приходится разбираться в механике на ходу. Такое дизайнерское решение обусловлено традициями серии Soulsborne.

Доступность

Сразу укажите переназначение клавиш, субтитры, масштаб интерфейса, режимы удержания/переключения, настройки тряски камеры, цветовые дублирующие сигналы и паузу. Не кодируйте критическую информацию только цветом или звуком.

Критерий UX. Новый игрок после события способен своими словами объяснить причину и назвать следующее доступное действие.

8. Как связать GDD с Unity 3D

GDD не должен диктовать архитектуру, но обязан разделять правила, данные и представление. Это позволяет менять баланс без переписывания логики и не смешивать дизайнерские параметры со случайными числами в коде.

Соответствие «Дизайн - Unity»

Сущность в документе — Модель + анимации + набор компонентов, получаем Enemy.prefab в Unity

Параметры противника — ScriptableObject: EnemyArchetypeData, в виде компонента Enemy.prefab

Состояния — Patrol, Suspicious, Alert как текущие значения переменных компонента Enemy.prefab.

События — Реакции на определённые действия игрока или его появление, которые хранят скрипты компонентов.

Уровень — Сцена которая загружена сейчас. Например GothicCastle01

Правила столкновения — каким образом расчитываются столкновения префаба enemy с игровым миром, проходит ли он сквозь все объекты или только через некоторые.

Сохранение — стабильный ID + сериализуемые данные. Каждая сущность обладает набором параметров которые могут изменяться. Иногда нужно их сохранять, не обнуляя при повторной загрузке уровня.

Пример структуры данных оружия

Скрипт **WeaponData**: ID, displayName, damage, fireRate, range, spreadMin, spreadMax, recoilProfile, magazineSize, reloadTime, ammoType, audioSet, vfxSet, animationSet.

В GDD рядом с каждым полем укажите единицы измерения и допустимый диапазон. Например, fireRate — выстрелов в секунду, а не «скорость 8».

Соглашения проекта

Единицы: метры, секунды, градусы, единицы ресурса.

Имена: префиксы только если они помогают поиску; ID не зависят от локализованного текста.

Сцены: bootstrap, меню, gameplay, additive lighting/streaming — назначение фиксируется.

Слои и теги: владелец и таблица использования; избегайте строковых сравнений в логике.

Debug: отображение состояний ИИ, зон слуха, урона, точек появления и событий квеста.

Практика. Для каждой механики выделите отдельно: данные, логику, визуальное представление и звук. Если изменение урона требует правки Animator или префаба, структуру системы стоит пересмотреть.

9. Производственный документ и управление изменениями

Дизайн становится производимым, когда каждое решение связано с задачей, зависимостью и критерием готовности. GDD объясняет правило, а трекер задач отвечает, кто и когда его реализует.

Статусы задач

Idea — не подтверждена ценностью - короткая гипотеза.
Prototype — проверяется основной риск - играбельный серый прототип.
Approved — решение принято - актуальная спецификация.
Production — создаются код и контент - задачи по дисциплинам.
Test — проверяются критерии - отчёт и список дефектов.
Done — соответствует Definition of Done - сборка + документация.
Cut / Archive — не входит в текущий объём - причина и условия возврата.

Журнал решений

Заполняется по мере их принятия, для того что бы можно было вернуться и узнать почему механика работает именно так, а не как в изначальном дисдоке.

Журнал должен описывать принятые решения, когда они были сделаны, почему и какие действия повлекут.

Например:

12.03 - сократить окно парирования до 0,16 с, потому что это слишком универсальная защита. Повторить тест на новичках

18.03 - убрать мини-карту из стелс-миссии, потому что она снижает наблюдение за миром. Добавить читаемые ориентиры.

Definition of Done для механики

Работает на целевых устройствах ввода и в заявленных состояниях персонажа.
Параметры вынесены в данные, имеют единицы и безопасные ограничения.
Есть визуальная, звуковая и интерфейсная обратная связь по спецификации.
Пройдены happy path, крайние случаи, сохранение/загрузка и смена сцены.
Отладочные средства и аналитические события доступны команде.
GDD, таблица баланса и список известных ограничений обновлены.

Контроль объёма: Для каждой механики, цели или функции, укажите приоритет.
Must — должно быть точно, высокий приоритет.
Should — должно быть, но не так критично как Must, приоритет средний.
Could — можно добавить, не критично. Низкий приоритет.
Wont — низкий приоритет, обычно это пункты из backlog.
Если всё имеет приоритет **Must**, то приоритетов нет.

10. Прототипирование, плейтесты и метрики

Прототип отвечает на один риск с минимальной стоимостью. Он не обязан быть красивым, но должен давать достоверное поведение: нужные тайминги, размеры, скорость и обратную связь.

Формула гипотезы

«Мы считаем, что [игрок] в ситуации [контекст] выберет/поймёт [поведение], потому что [причина]. Мы признаем гипотезу подтверждённой, если [наблюдаемый критерий]».

Пример: новый игрок заметит безопасный обход охранника благодаря движущейся тени. Критерий: 4 из 5 участников находят маршрут без подсказки за 90 секунд.

Протокол теста

Зафиксируйте версию сборки, аудиторию и проверяемые вопросы.

Не объясняйте управление сверх подготовленного вводного текста.

Наблюдайте действия, паузы, повторы и ошибочные ожидания; не обучайте во время задания.

После сессии задайте нейтральные вопросы: «Что вы ожидали?», «Почему выбрали этот путь?».

Разделите факт, интерпретацию и предлагаемое изменение.

Изменяйте одну крупную переменную за итерацию и повторяйте тест.

Полезные метрики

Бой - время до победы, полученный урон, расход лечения - даст нам понять почему бой скучен или нечитаем.

Стелс - число тревог, время в поиске, маршруты даёт понимание осознанности выбора.

Уровень - время участка, смерти, развороты - подсветит нам эмоцию и понимание цели.

Экономика - источники/стоки, остаток ресурса, покажет ценность решения для игрока.

Правило. Метрика показывает место вопроса, но редко даёт ответ без наблюдения и разговора.

11. Жанровый пример: Souls-like — концепт

Учебный проект «Пепельный Предел»

Однопользовательская action RPG от третьего лица. Игрок исследует компактный взаимосвязанный район, изучает паттерны противников, рискует накопленным ресурсом и открывает короткие пути. MVP: 30–45 минут, один хаб, один маршрут с развилкой, три обычных противника и один босс.

Столпы

Ответственная атака: каждое действие имеет начало, активную фазу и восстановление; спам создаёт риск.

Смерть обучает: потеря возвращаема, путь назад короче после освоения пространства.

Читаемая угроза: силуэт, звук и подготовка атаки позволяют понять ответ до получения урона.

Мир складывается в карту: ориентиры и короткие пути важнее маркеров.

Циклы

Масштабы цикла

Секунды — прочитать телеграфирование противника > выбрать защиту > найти окно > атаковать > восстановить позицию.

Минуты — войти на локацию > изучить угрозу > победить/обойти > получить ресурс > открыть путь.

Часы — Изучить всю локацию > улучшить оружие > победить босса локации > открыть следующую локацию.

Риск и возврат

После гибели весь не оставленный в хранилище ресурс, остаётся в точке последней смерти. Игрок может вернуть его одним касанием. Повторная гибель до возврата удаляет ресурс. Точки отдыха восстанавливают здоровье, выносливость и обычных врагов, но открытые двери и побеждённые боссы сохраняются.

Границы MVP

Нет сетевой игры, вторжений, процедурной генерации, нескольких классов и сложной ветвящейся истории. Один набор оружия проверяет главное: вес атаки, чтение противника, ценность уклонения и понятный возврат после смерти.

Главный риск прототипа. Бой может ощущаться тяжёлым не из-за осмысленных обязательств, а из-за задержки ввода. Проверяйте input buffering и момент начала реакции отдельно.

Souls-like — спецификация боя и встречи

Системная таблица

Выносливость: атаки и уклонение её расходуют; бег — только вне боя у персонажа 100 ед.; реген 18/с после задержки 0,6 с

Захват цели: приоритет видимости и угла к камередальность 18 м; разрыв 24 м

Poise врага: сильные удары накапливают пошатывание, порог 60; восстановление через 2 с.

Лечение: ограниченные заряды, обязательная анимация, 3 заряда; 45 HP; применение 1,2 с.

Парирование: только помеченные атаки, окно 0,16 с; сильный ответ при удачном отражении атаки.

Пример врага: Пепельный страж

Роль: обучить позднему уклонению и атаке в спину.

Дистанция атаки: 0–2 м: связка из двух ударов; 2–5 м: медленный выпад; дальше 5 м: сокращает дистанцию, не атакуя.

Выпад: Перед выпадом оружие уходит назад на 0,55 с и звучит металлический скрип. После промаха Recovery 0,9 с.

Имеет щит, блокирует фронтальный урон, но не удар в спину.

Учебная встреча

Игрок видит стража со спины и может нанести критический удар: безопасная демонстрация.

В коридоре дальше, тот же враг встречается лицом; рядом колонна для контроля дистанции.

Далее, на открытой площадке, страж комбинируется с быстрым слабым врагом: проверка приоритетов.

За площадкой находится короткий путь к точке отдыха: награда за освоение.

Приёмочные тесты

Телеграф выпада различим без интерфейсного маркера минимум у 4 из 5 тестеров.

Камера не проходит сквозь стену и сохраняет угрозу в кадре при захвате цели.

Возврат к месту смерти от контрольной точки занимает не более 30 секунд.

Победа возможна стартовым оружием без улучшений; среднее время боя после изучения — 0,5 - 1 минута.

12. Жанровый пример: стелс-игра — концепт

Учебный проект «Тихий Контур»

Камерная стелс-игра от третьего лица. Игрок проникает в исследовательский комплекс, крадёт данные и выходит, избегая прямого столкновения. Главный крючок: электрические цепи уровня можно временно перенаправлять, управляя светом, дверями и камерами.

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, купив полную легальную версию на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.